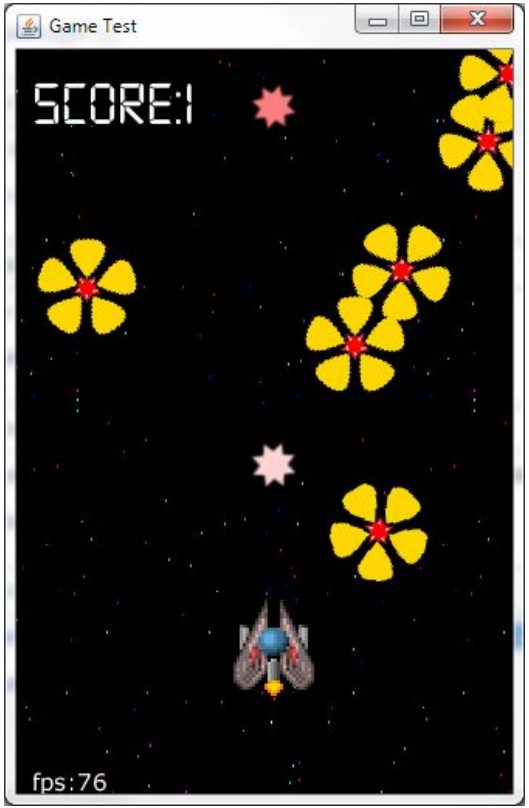
Scene2D

Actors

1. What is Scene2D.

Is a 2D scene graph for building applications and UIs using a hierarchy of actors that provides the following features:

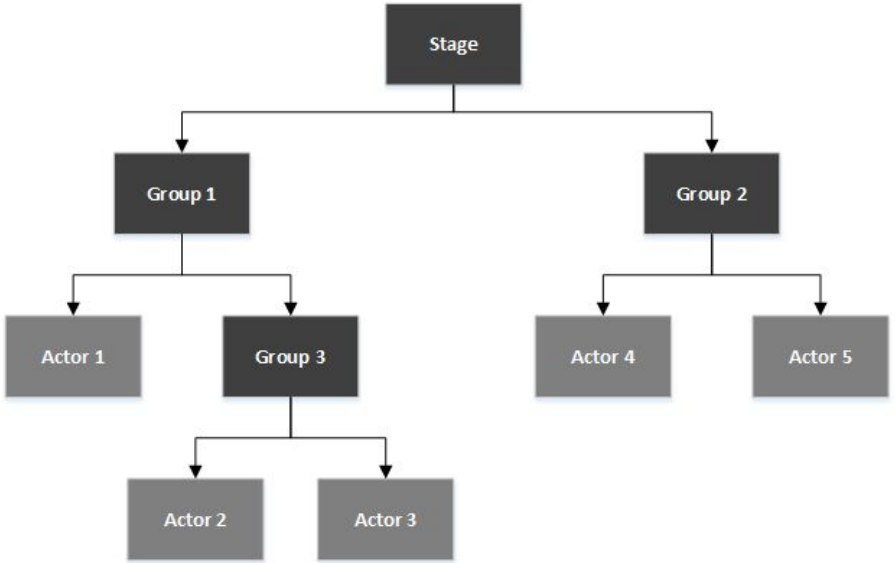
- Rotation and scale of a group is applied to all child actors.
- Simplified 2D drawing via SpriteBatch.
- Hit detection of rotated and scaled actors.
- Routing of input and other events to the appropriate actor.



1. What is Scene2D.

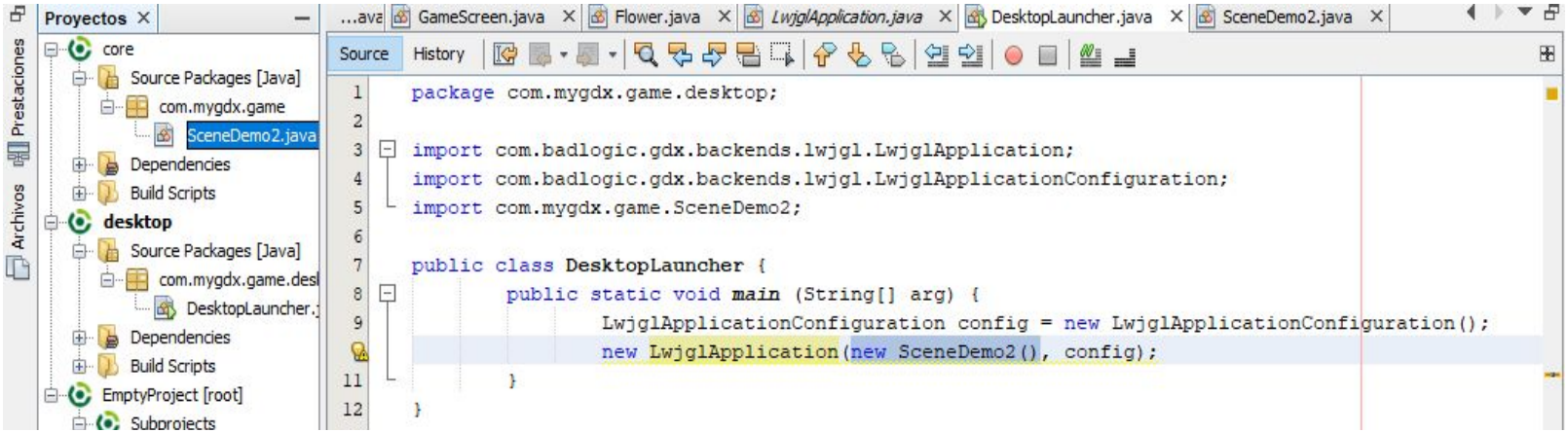
Scene2d has three classes at its core:

- The Stage class has a camera, SpriteBatch, and a root group and handles drawing the actors and distributing input events.
- The Group class is an actor that may have child actors.
- The Actor class is a node in the graph which has a position, rectangular size, origin, scale, rotation, and color.



2. Explain in code

In this example we're going to create the the class that contains Scene class as the main of the core.



2.1. Scene

- Create method:
  - Create Stage object.
  - Create actions class and assign values:
    - MoveToAction
    - RotateToAction
    - ScaleToAction
  - Add action in Stage.

```
private Stage stage;

@Override
public void create() {
    stage = new Stage();
    Gdx.input.setInputProcessor(stage);

    MyActor myActor = new MyActor();
    //1.myActor.setTouchable(Touchable.enabled);

    MoveToAction moveAction = new MoveToAction();
    moveAction.setPosition(300f, 0f);
    moveAction.setDuration(10f);
    myActor.addAction(moveAction);

    stage.addActor(myActor);
}
@Override
public void dispose() {
}
@Override
public void render() {
    Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
    stage.act(Gdx.graphics.getDeltaTime());
    stage.draw();
}

@Override
public void resize(int width, int height) {
}
@Override
public void pause() {
}
@Override
public void resume() {
}
```

2.1. Scene

- Render method:
  - Assign the background color
  - Assign the time since the previous frame.
  - draw the frame

```
private Stage stage;

@Override
public void create() {
    stage = new Stage();
    Gdx.input.setInputProcessor(stage);

    MyActor myActor = new MyActor();
    //1.myActor.setTouchable(Touchable.enabled);

    MoveToAction moveAction = new MoveToAction();
    moveAction.setPosition(300f, 0f);
    moveAction.setDuration(10f);
    myActor.addAction(moveAction);

    stage.addActor(myActor);
}
@Override
public void dispose() {
}
@Override
public void render() {
    Gdx.gl.glClear(GL20.GL_COLOR_BUFFER_BIT);
    stage.act(Gdx.graphics.getDeltaTime());
    stage.draw();
}

@Override
public void resize(int width, int height) {
}
@Override
public void pause() {
}
@Override
public void resume() {
}
```

2.2. Actor

In the main class i create an Actor class.

- Create a texture object
- Constructor:
  - Assign the size
  - Assign initial position
- Draw method
  - Assign parameters for actor
- Act Method
  - Run the actions created in Scene

```
public class SceneDemo2 implements ApplicationListener {

    public class MyActor extends Actor {

        Texture texture = new Texture("correcaminos.png");
        float actorX = 0, actorY = 0;
        public boolean started = false;

        public MyActor() {
            setBounds(getX(), getY(), texture.getWidth(), texture.getHeight());
        }
        @Override
        public void draw(Batch batch, float alpha) {
            batch.draw(texture, this.getX(), getY(), this.getOriginX(), this.getOriginY(), this.getWidth(),
                    this.getHeight(), this.getScaleX(), this.getScaleY(), this.getRotation(), 0, 0,
                    texture.getWidth(), texture.getHeight(), false, false);
        }
        @Override
        public void act(float delta) {
            for (Iterator<Action> iter = this.getActions().iterator(); iter.hasNext();) {
                iter.next().act(delta);
            }
        }
    }
```